THE MATHEMATICS OF ADAPTIVE DISTRIBUTED CONTROL

David H. Wolpert

NASA Ames Research Center

http://ti.arc.nasa.gov/~dhw/

NASA-ARC-03-097

ROADMAP

1) ***What is distributed control, formally?***

2) ***Review constrained optimization***

3) ***Optimal control policy for distributed agents***

4) ***How to find that policy in a distributed way***

5) ***A movie: "PC in the real world"***

DISTRIBUTED ADAPTIVE CONTROL

1) ***Control of routers in a network.***
2) ***Control of robots working together to construct a spacestation.***
3) ***Control of flaplets on an aircraft wing.***
4) ***Control of signals to human teams performing a joint task.***
5) ***Control of variables in a parallel computer algorithm to optimize a function.***

Must be adaptive (i.e., not wed to a system model) to

i) ***Avoid brittleness;***

ii) ***Scale well;***

iii) ***Be fault-tolerant;***

iv) ***Be widely applicable, with minimal (or even no) hand-tuning.***

THE GOLDEN RULE

DO NOT:

Find a value of a variable x, that optimizes a function

INSTEAD:

Find a distribution over x, that optimizes an expectation value

ADVANTAGES

1) ***Works for arbitrary (mixed) data types x - P(x) is always a vector of real numbers, no matter what data type x is.***

2) ***So in particular, leverages techniques for the optimization for Euclidean vectors - the most powerful optimization techniques we have. ("Gradient descent for symbolic variables.")***

3) ***P(x) provides sensitivity information (which components of x are most important).***

MORE ADVANTAGES

4) *Can be "seeded" with solutions of other algorithms: peaks of initial P(x).*
5) *Can include Bayesian prior knowledge.*
6) *Automatically accomodate noisy, poorly modeled problems.*

- *Deep connections with statistical physics and game theory. So*
  - *Especially suited for distributed domains.*
  - *Especially suited for very large problems.*

WHAT IS DISTRIBUTED CONTROL?

1) ***A set of N agents: Joint move $x = (x_1, x_2, ..., x_N)$***

2) ***Since they are distributed, their joint probability is a product distribution:***

$$\boxed{\mathbf{q(x) \;=\; \prod_i q_i(x_i)}}$$

- ***This definition of distributed agents is adopted from (normal form) noncooperative game theory.***

EXAMPLE: KSAT

- **$x = \{0, 1\}^N$**

- **A set of many disjunctions, "clauses", each involving K bits.**
  E.g., $(x_2 \vee x_6 \vee \sim x_7)$ is a clause for K = 3

- **Goal: Find a bit-string x that simultaneously satisfies all clauses. G(x) is #violated clauses.**

- **For us, this goal becomes: find a $q(x) = \prod_i q_i(x_i)$ tightly centered about such an x.**

The canonical computationally difficult problem

ROADMAP

1) *What is distributed control, formally?*

2) *Review constrained optimization*

3) *Optimal control policy for distributed agents*

4) *How to find that policy in a distributed way*

5) *A movie: "PC in the real world"*

REVIEW OF CONSTRAINED OPTIMIZATION

1) **We want to minimize a smooth function $f(y \in \Re^n)$ subject to K equality constraints $\{h_i(y) = 0\}$.**

2) **Example: Each $h_i(q)$ says a subset of q's components sum to 1, i.e., q is a probability distribution.**

3) **Define** $\boxed{L(\{\lambda_i\}, y) \equiv f(y) + \sum_i \lambda_i h_i(y)}$

4) ***L* is the *Lagrangian*, and $\{\lambda_i\}$ the *Lagrange parameters*.**

REVIEW OF CONSTRAINED OPTIMIZATION - 2

4) In general (finite gradients), the solution is a critical point of *L*, i.e., it is the y value at the point

$$\max_{\{\lambda_i\}} \min_y L(\{\lambda_i\}, y)$$

5) To find the solution, solve

$$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial \lambda_i} = 0$$

REVIEW OF CONSTRAINED OPTIMIZATION - 3

6) Add inequality constraints: together with equality constraints they restrict y to a *feasible region* $\subset \Re^n$.

7) In special cases (e.g., convex problems) can deal with inequality constraints by adding Lagrange parameter terms to L.

REVIEW OF CONSTRAINED OPTIMIZATION - 4

8) More general approach: add a *barrier function* ϕ_j to L for each inequality constraint j:

$$L(\{\lambda_i\}, \{c_j\}, y) \equiv f(y) + \sum_i \lambda_i h_i(y) + \sum_j c_j \, \phi_j(y)$$

9) Each ϕ_j is non-negative, and infinite if the j'th inequality constraint is violated.

10) Each *barrier parameter* c_j is non-negative, and gets reduced to 0 via annealing.

ROADMAP

1) *What is distributed control, formally?*

2) *Review constrained optimization*

3) *Optimal control policy for distributed agents*

4) *How to find that policy in a distributed way*

5) *A movie: "PC in the real world"*

ITERATIVE DISTRIBUTED CONTROL

(P1) Find $$\min_{\{q_i\}} \int dx\ \ G(x)\prod_i q_i(x_i)$$

such that

$$\forall i, \int dx_i\ q_i(x_i)\ =\ 1,\ \forall x_i,\ q_i(x_i) \geq 0$$

- **A constrained optimization problem with both equality and inequality constraints.**

ITERATIVE DISTRIBUTED CONTROL - 2

(P2) Find the $\{q_i\}$ minimizing

$$\int dx \;\; G(x) \prod_i q_i(x_i) + \sum_i \int dx_i \;\; c(i, x_i) \phi_i(q_i(x_i))$$

such that $$\forall i, \int dx_i \;\; q_i(x_i) \;\; = \;\; 1$$

- **A common barrier function is** $\phi_i(y) = y \ln[y]$
- **If also all c_i = T, then the objective function of (P2) is the *free energy*,** $F_T(q) \;=\; E_q(G) - TS(q)$

AUTOMATED ANNEALING

1) **Ultimately want $T \rightarrow 0$, starting at high T.**
2) **So want to minimize $F_T(q)$ over *both T and q*.**
3) **Can use gradient descent to do this.**
4) **$\partial F / \partial q$ components of gradient discussed below.**
5) $\boxed{\partial F / \partial T \;=\; \partial [E_q(G) - TS(q)] / \partial T \;=\; -S(q).}$
6) **So for fixed descent stepsize, ΔT is given by the ratio of $-S(q)$ to $\partial F / \partial q$.**
7) **In particular, $|\Delta T|$ shrinks as $S(q)$ does, i.e., as the optimization progresses.**

KULLBACK-LEIBLER DISTANCE AND FREE ENERGY

1) **The *Kullback-Leibler* (KL) distance between probability distributions a(y) and b(y) is**

$$\mathbf{KL(a \,||\, b) \;=\; -\!\int\! dy\; a(y)\, \ln[b(y) / a(y)]}$$

2) **The *Boltzmann distribution* is** $\mathbf{p_\beta(x) \propto e^{-\beta G(x)}}$

As $\beta \to \infty$, $p_\beta(x)$ gets peaked about $\mathrm{argmin}_x G(x)$

3) **Let $T = 1/\beta$:** $\mathbf{KL(q \,||\, p_\beta) \;=\; F_T(q).}$

Minimizing $F_T(q)$ minimizes distance to the Boltzmann distribution.

EXAMPLE: KSAT

1) $S(q) = -\sum_i [b_i \ln(b_i) + (1 - b_i) \ln(1 - b_i)]$

where b_i is $q_i(x_i = \text{TRUE})$

2) $E_q(G) = \sum_{clauses\ j,\ x} q(x)\, K_j(x)$

$= \sum_{clauses\ j,\ x,\ i} \prod_i q_i(x_i)\, K_j(x)$

where $K_j(x) = 1$ iff x violates clause j

Our algorithm: i) Find q minimizing $E_q(G) - TS(q)$;

ii) Lower T and return to (i).

ROADMAP

1) *What is distributed control, formally?*

2) *Review constrained optimization*

3) *Optimal control policy for distributed agents*

4) *How to find that policy in a distributed way*

5) *A movie: "PC in the real world"*

GRADIENT DESCENT OF $F_T(q)$

1) **Each i works to shrink $F_T(q_i, q_{(i)})$ using only partial information of the other agents' distribution, $q_{(i)}$.**

2) **The $q_i(x_i)$ component of $\nabla F_T(q)$, restricted to the space of allowed $q_i(x_i)$, is**

$$\boxed{\begin{array}{c} E_{q_{(i)}}(G \mid x_i) \;+\; T \ln[q_i(x_i)] \\ - \\ (1/|X_i|) \int dx'_i \, [E_{q_{(i)}}(G \mid x'_i) \;+ T \ln[q_i(x'_i)]] \end{array}}$$

where $E_{q_{(i)}}(G \mid x_i)$ is expected G given x_i.

***GRADIENT DESCENT* - 2**

3) Each agent i knows its values of $\ln[q_i(x_i)]$.

4) Say each agent i knows the $E_{q_{(i)}}(G \mid x_i)$.

Each q_i knows how it should

change under gradient descent over $F_T(q)$

5) Similarly the Hessian can readily be estimated (for Newton's method), etc.

BROUWER UPDATING TO FIND q

1) Solve for the q minimizing F(q):

$$q_i(x_i) \propto e^{-\beta E_{q_{(i)}}(G|x_i)}$$

where again, $E_{q_{(i)}}(G \mid x_i)$ is expected G given x_i, when other agents are distributed according to $q_{(i)}$

2) When each agent i knows/estimates $E_{q_{(i)}}(G \mid x_i)$, they can simultaneously jump to their optimal q_i.

This is ***Parallel Brouwer Updating***

PARALLEL BROUWER UPDATING

1) **Related to game theory's "ficticious play", and to some reinforcement learning algorithms.**

2) **Can have slow convergence.**

> **The problem is that each agent does what would be optimal *if the other agents didn't change their distributions*. But they *do* change.**

3) **Parallel Brouwer can even worsen the Lagrangian in any given update.**

SERIAL BROUWER UPDATING

1) **Instead, can cycle through which agent Brouwer updates round robin.**

2) **Can cycle through which agent Brouwer updates randomly.**

3) **Either can have slow convergence, when there are many agents.**

4) **However with any kind of serial Brouwer, every update by an agent improves the Lagrangian.**

GREEDY SERIAL BROUWER

1) The *Lagrangian gap* of agent i is the drop in $F_T(q)$ if only i updates. With $N_{i,q}(G)$ defined as i's normalization constant, the gap equals

$$\ln[N_{i,q}(G)] \; + \; E_{q_i}(E(G \mid x_i)) \; + \; S_i(q_i)$$

2) The agent with the largest gap updates.

Mixed serial/parallel Brouwer updating :

Optimal Stackelberg game, i.e., optimal organization chart

EXAMPLE: KSAT

1) Evaluate $\boxed{E_{q_{(i)}}(G \mid x_i)}$ - the expected number of violated clauses if bit *i* is in state x_i - for every i, x_i

2) In gradient descent, decrease each $q_i(x_i)$ by

$$\boxed{\alpha[E_{q_{(i)}}(G \mid x_i) + T \ln[q_i(x_i)] \; - \; \text{const}_j]}$$

where α is the stepsize, and const_j is an easy-to-evaluate normalization constant.

3) We actually have a different T for each clause, and adaptively update all of them.

ADAPTIVE DISTRIBUTED CONTROL

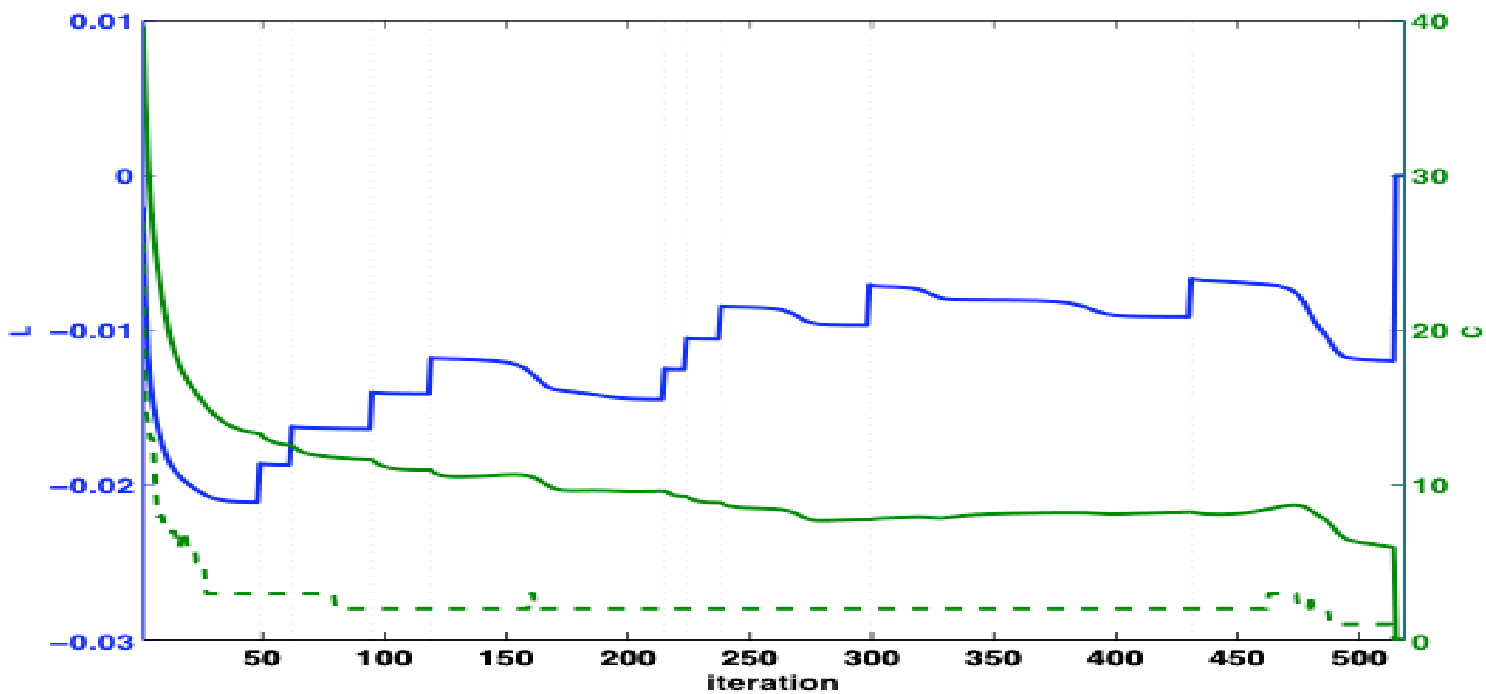
1) In *adaptive* control, don't know functional form of G(x). So use Monte Carlo:

- **Sample G(x) repeatedly according to q;**
- **Each i independently estimates $E_{q_{(i)}}(G \mid x_i)$ for all its moves x_i.**

So each q_i can adaptively estimate its update

EXAMPLE: KSAT

i) Top plot is Lagrangian value vs. iteration;
ii) Middle plot is average (under q) number of constraint violations;
iii) Bottom plot is mode (under q) number of constraint violations.



CONCLUSION

1) *A distributed system is governed by a product distribution q, by definition.*

2) *So distributed adaptive control is adaptive search for the q that optimizes $E_q(G)$.*

3) *That search can be done many ways, e.g., gradient descent, with or without Monte Carlo sampling.*